### Zoznam skratiek, značiek a symbolov

Klient

Paket

Server

Tickrate

Tick

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### Visual Studio

Visual Studio je populárne integrované vývojové prostredie (IDE) vyvinuté spoločnosťou Microsoft. Je určené pre vývoj aplikácií na rôznych platformách, ako sú Windows, macOS, Android, iOS či webové aplikácie. Vzniklo s cieľom zjednodušiť a zefektívniť proces vývoja softvéru, pričom ponúka širokú paletu nástrojov pre programátorov.

Prvá verzia Visual Studia bola uvedená na trh v roku 1997 pod názvom Microsoft Visual Studio 97. Išlo o balík nástrojov, ktorý združoval rôzne programovacie jazyky, ako Visual Basic, Visual C++, Visual FoxPro a ďalšie. Toto spojenie malo za cieľ zjednotiť rôzne vývojové nástroje do jedného prostredia.

V priebehu rokov Visual Studio prešlo viacerými vylepšeniami. Medzi najvýznamnejšie míľniky patrí:

* **2002**: Uvedenie podpory pre .NET Framework, čím sa stalo hlavným IDE pre vývoj aplikácií v jazykoch C# a VB.NET.
* **2010**: Zlepšenie používateľského rozhrania a zavedenie funkcií, ako sú IntelliSense (automatické dopĺňanie kódu) a podpora pre cloudové služby.
* **2017**: Rozšírenie podpory na vývoj pre Linux a macOS, čím sa IDE stalo atraktívnejším pre multiplatformový vývoj.

Visual Studio ponúka širokú škálu nástrojov, vrátane:

* **IntelliSense**: Inteligentné návrhy kódu, ktoré pomáhajú programátorom rýchlejšie písať a odstraňovať chyby.
* **Debugging**: Pokročilé nástroje na ladenie a sledovanie aplikácií.
* **Git integrácia**: Jednoduché spravovanie verzií kódu pomocou Git a GitHub.
* **Rozšíriteľnosť**: Možnosť pridať tisíce rozšírení cez Visual Studio Marketplace.

V súčasnosti je Visual Studio dostupné v niekoľkých edíciách, ako napríklad **Community** (bezplatná verzia pre individuálnych vývojárov a malé tímy), Professional a Enterprise (pre väčšie organizácie). Od roku 2015 je k dispozícii aj Visual Studio Code, ľahšia a multiplatformová verzia, ktorá si získala veľkú popularitu medzi vývojármi.

### MonoGame

MonoGame je multiplatformová open-source framework knižnica, ktorá umožňuje vývoj hier pre rôzne platformy, ako sú Windows, macOS, Linux, Android, iOS, PlayStation, Xbox a ďalšie. Ide o nástroj určený predovšetkým pre nezávislých vývojárov, ktorí chcú vytvárať hry v **C#** a .NET ekosystéme.

MonoGame vzniklo v roku 2009 ako projekt na reimplementáciu Microsoft XNA Framework, ktorý bol populárnym frameworkom na vývoj hier pre Xbox 360 a Windows. Po tom, čo Microsoft ukončil podporu XNA v roku **2013**, MonoGame prevzalo štafetu a umožnilo vývojárom pokračovať v používaní podobných nástrojov, no s podporou širšieho spektra platforiem.

Projekt bol pôvodne vytvorený Josem Antonio Lealom de Figueirom a stal sa otvoreným projektom s príspevkami od komunity. MonoGame si získalo obľubu najmä vďaka svojej jednoduchosti, multiplatformovosti a flexibilite.

#### **Funkcie a možnosti**

MonoGame ponúka základné nástroje na tvorbu hier, vrátane:

* **Podpory 2D a 3D hier**: Možnosti pre jednoduché alebo komplexné herné projekty.
* **Multiplatformový vývoj**: Kód napísaný v MonoGame možno jednoducho portovať na rôzne zariadenia a operačné systémy.
* **Grafika a zvuk**: Podpora pre spracovanie grafiky, animácií a zvuku.
* **Open-source povaha**: Vývojári môžu prispievať a prispôsobovať knižnicu svojim potrebám.

MonoGame bolo použité na vývoj viacerých známych hier, medzi ktoré patria:

1. **Stardew Valley** (2016): Kultová farming RPG hra vytvorená Ericom Baronom (ConcernedApe). Táto hra, ktorá si získala milióny fanúšikov, je jedným z najväčších úspechov MonoGame.
2. **Terraria** (porty): Populárna sandboxová hra bola čiastočne prenesená na iné platformy pomocou MonoGame.
3. **Barotrauma** (2019): Simulátor ponorky pre viacerých hráčov, odohrávajúci sa na, Jupiterovom mesiaci, Európa.

MonoGame je stále aktívne vyvíjané komunitou a slúži ako základ pre mnohých nezávislých vývojárov. Je to výborná voľba pre tých, ktorí hľadajú stabilný, flexibilný a multiplatformový framework na vývoj hier, najmä ak uprednostňujú jazyk C#.

### Clip Studio Paint

Clip Studio Paint (pôvodne známy ako Manga Studio) je profesionálny softvér na digitálne kreslenie a maľbu, ktorý si získal popularitu medzi umelcami po celom svete. Je ideálny pre tvorbu ilustrácií, komiksov, manga, animácií a konceptuálneho umenia. Vďaka svojej všestrannosti a pokročilým nástrojom je Clip Studio Paint využívaný amatérmi aj profesionálmi v rôznych kreatívnych odvetviach.

Clip Studio Paint bol vyvinutý japonskou spoločnosťou CELSYS a prvýkrát uvedený na trh v roku 2001 pod názvom Comic Studio. Spočiatku bol zameraný najmä na tvorbu manga a komiksov, čo ho odlíšilo od iných grafických programov, ako sú Adobe Photoshop alebo Corel Painter.

V roku 2012 došlo k premenovaniu na Clip Studio Paint a výraznému rozšíreniu jeho funkcií. Nástroj sa stal dostupným nielen pre tvorcov manga, ale aj pre digitálnych ilustrátorov a animátorov.

Od svojho vzniku prešiel Clip Studio Paint mnohými aktualizáciami, ktoré pridali nové funkcie, vrátane:

* **Podpora 3D modelov**: Pre zjednodušenie vytvárania póz a perspektív.
* **Animácia**: Možnosť vytvárať 2D animácie priamo v programe.
* **Cloudová synchronizácia**: Umožňuje užívateľom pracovať na viacerých zariadeniach.

Clip Studio Paint je známy pre svoje špecializované nástroje, medzi ktoré patria:

* **Prispôsobiteľné štetce**: Program ponúka širokú škálu štetcov, ktoré je možné prispôsobiť podľa potrieb používateľa.
* **Nástroje na kreslenie komiksov**: Šablóny panelov, bubliny na text a špeciálne efekty pre komiksy.
* **Vektorové kreslenie**: Umožňuje jednoduché úpravy čiar a línií bez straty kvality.
* **Materiálová knižnica**: Zahŕňa textúry, pozadia a 3D modely, ktoré možno použiť vo svojich projektoch.

Clip Studio Paint je obľúbený medzi profesionálnymi komiksovými autormi, ilustrátormi a animátormi, a to najmä v Japonsku, kde je jedným z najpoužívanejších nástrojov na tvorbu manga. Medzinárodne si ho obľúbili aj indie tvorcovia, ktorí oceňujú jeho cenovú dostupnosť a špecializované funkcie.

Niektorí z významných umelcov a štúdií používajúcich Clip Studio Paint pochádzajú z oblasti mangy, komiksov a animácií, čím dokazuje svoju flexibilitu a profesionalitu.

Clip Studio Paint je dostupný pre rôzne platformy vrátane Windows, macOS, iOS, Android a Chromebook, čo umožňuje tvorbu na stolných počítačoch aj mobilných zariadeniach. V roku 2022 bol zavedený nový model predplatného pre cloudové funkcie, čím sa softvér stal prístupnejší pre globálne publikum.

### Git: Základný nástroj na správu verzií

Git je jedným z najpopulárnejších a najrozšírenejších nástrojov na správu verzií (version control system - VCS). Bol navrhnutý na efektívne sledovanie zmien v kóde, koordináciu práce medzi viacerými vývojármi a na zabezpečenie integrity dát. Git je open-source projekt, ktorý vytvoril Linus Torvalds v roku 2005, pôvodne pre správu vývoja jadra Linuxu.

#### **Čo je Git?**

Git je distribuovaný systém na správu verzií. To znamená, že každá kópia projektu obsahuje kompletnú históriu a stav repozitára (úložiska). Vďaka tomu sa Git líši od tradičných centralizovaných systémov, kde všetky zmeny závisia od jedného hlavného servera.

Hlavnou úlohou Gitu je:

* **Sledovanie zmien:** Git ukladá každú zmenu vykonanú v kóde, čo umožňuje vývojárom vrátiť sa k predošlým verziám.
* **Spolupráca:** Git umožňuje viacerým vývojárom pracovať na jednom projekte súčasne bez konfliktov.
* **Bezpečnosť:** Git používa kryptografické haše (SHA-1) na identifikáciu zmien, čo zaručuje, že dáta nemôžu byť neúmyselne alebo úmyselne zmenené.

#### **Ako Git funguje?**

Git funguje na princípe lokálnych, vzdialených a staging oblastí:

1. **Lokálny repozitár:** Každý vývojár má kópiu celého projektu na svojom počítači. Toto lokálne úložisko obsahuje všetky zmeny a históriu projektu.
2. **Staging area:** Pred odoslaním zmien do hlavného repozitára sa zmeny pridávajú do staging oblasti (pomocou príkazu **git add**). Táto oblasť umožňuje vývojárom skontrolovať, ktoré zmeny sa majú uložiť.
3. **Commit:** Po pridaní zmien do staging oblasti vývojár uloží zmeny do histórie projektu (pomocou príkazu **git commit**).
4. **Push a Pull:** Zmeny možno odoslať na vzdialený server (push) alebo stiahnuť najnovšie zmeny od ostatných vývojárov (pull).

Git sleduje každý krok vývoja projektu pomocou príkazov, ako sú:

* **git init:** Inicializácia nového Git repozitára.
* **git clone**: Klonovanie existujúceho repozitára.
* **git branch:** Práca s vetvami projektu.
* **git merge:** Spájanie zmien medzi vetvami.
* **git log:** Zobrazenie histórie zmien.

#### **Použitie Gitu**

Git sa používa v rôznych oblastiach vývoja softvéru a spolupráce. Medzi najbežnejšie využitia patria:

1. **Vývoj softvéru:**
   * Git umožňuje tímom efektívne spolupracovať na jednom projekte bez toho, aby si vzájomne prepisovali kód.
   * Práca s vetvami (branches) umožňuje vývojárom pracovať na nových funkciách alebo opravách chýb nezávisle od hlavného kódu (main/master branch).
2. **Správa projektov:**
   * Git trackuje všetky zmeny, čo je užitočné pre sledovanie pokroku v projekte a identifikáciu autorov zmien.
3. **Zálohovanie a obnova:**
   * Git slúži ako spoľahlivý spôsob zálohovania kódu a obnovenia predošlých verzií.

#### **Git a aplikácie**

Na používanie Gitu môžu vývojári pracovať buď priamo cez príkazový riadok, alebo cez grafické používateľské rozhrania (GUI). Medzi najznámejšie aplikácie na správu repozitárov patria:

1. **GitHub:**
   * Najpopulárnejšia platforma na hosťovanie repozitárov.
   * Poskytuje nástroje na spoluprácu, ako sú pull requesty, issues a diskusie.
   * Podporuje open-source projekty a súkromné repozitáre.
2. **GitLab:**
   * Alternatíva k GitHubu s dôrazom na DevOps a CI/CD nástroje.
   * Umožňuje vlastné hosťovanie a integráciu pokročilých workflowov.
3. **Bitbucket:**
   * Populárna medzi vývojármi využívajúcimi Atlassian ekosystém (napr. Jira).
   * Podporuje Git aj Mercurial repozitáre.
4. **SourceTree:**
   * Bezplatná desktopová aplikácia na správu Git repozitárov s intuitívnym grafickým rozhraním.
5. **GitKraken:**
   * Výkonný nástroj s prehľadným rozhraním a funkcionalitou na správu vetiev a riešenie konfliktov.

#### Git dnes

Git je dnes základným nástrojom vo svete softvérového vývoja. Používa sa v malých aj veľkých tímoch, od startupov až po technologické giganty. Vďaka svojej flexibilite a výkonnosti je Git preferovaným systémom na správu verzií pre projekty akéhokoľvek rozsahu.

### História počítačových hier pre viacerých hráčov

Multiplayerové hry sa stali neoddeliteľnou súčasťou herného priemyslu a formovali spôsob, akým hráči po celom svete komunikujú a súťažia. Ich vývoj siaha až do 50. rokov 20. storočia a postupne sa premenil na jeden z najdôležitejších segmentov v histórii hier.

#### **Počiatky multiplayerových hier (50. – 70. roky)**

Prvé pokusy o multiplayerové hry sa objavili už v počiatkoch počítačového výskumu:

* **1958 –** Tennis for Two: Jednou z prvých hier, ktoré umožnili interakciu dvoch hráčov, bola hra Tennis for Two, ktorú vytvoril americký fyzik William Higinbotham. Bežala na osciloskope a simulovala tenisový zápas, pričom dvaja hráči ovládali odrazenie loptičky cez sieť.
* **1962 –** Spacewar!: Vytvorená na univerzite MIT tímom vedeným Stevom Russellom, Spacewar! bola jednou z prvých hier, ktoré umožňovali multiplayer. Dvaja hráči ovládali vesmírne lode, ktoré bojovali v prostredí simulovaného vesmíru.

Tieto hry boli obmedzené na univerzitné výskumné počítače a nikdy sa nedostali do širokej verejnosti. Avšak ukázali, že počítačové hry môžu byť sociálnym zážitkom, čo pripravilo cestu pre ďalší vývoj.

#### **Zrodenie online multiplayerových hier (70. – 80. roky)**

So vznikom sietí a výpočtových systémov sa otvorili nové možnosti pre multiplayerové hranie:

* **1973 –** PLATO systémy: Vzdelávacie systémy PLATO vyvinuté na Univerzite Illinois umožnili vývoj niekoľkých multiplayerových hier, ako Empire a Spasim. Empire bola strategická hra pre viac hráčov, zatiaľ čo Spasim (1974) bola prvá 3D multiplayerová strieľačka.
* **1978 –** Multi-User Dungeon (MUD): Prvým veľkým krokom k moderným online hrám bolo vytvorenie MUD, textovej hry, ktorá kombinovala adventúru a RPG prvky. MUD bol vyvinutý Royom Trubshawom a Richardom Bartleom na univerzite v Essexe. Táto hra umožňovala hráčom spolupracovať a bojovať v textovom virtuálnom svete, čím vznikol základ pre MMORPG.

V tomto období sa multiplayerové hry začali vyvíjať z lokálnych zážitkov do online sveta, umožňujúceho interakciu hráčov na rôznych miestach.

#### **Zlatý vek arkád a LAN hier (80. – 90. roky)**

V 80. rokoch sa multiplayerové hry rozšírili aj do arkádových hier:

* **1980 –** Pac-Man (a jeho multiplayerové varianty): Arkádové hry ako Pac-Man, Galaga alebo Street Fighter umožnili hráčom súťažiť proti sebe priamo v herni. Rivalita medzi hráčmi bola jednou z hlavných atrakcií týchto hier.
* **Doom (1993)**: Príchod LAN (lokálnej siete) a FPS hry Doom od id Software priniesol revolúciu v multiplayerovom hraní. Doom umožnil hráčom spojiť svoje počítače a súťažiť v rýchlych multiplayerových zápasoch. Tento systém sa stal základom pre LAN party, ktoré boli populárne až do konca 90. rokov.

Okrem toho sa v tomto období začal rozmach stratégií pre viacerých hráčov, napríklad Command & Conquer (1995) alebo Warcraft II (1995), ktoré umožnili hráčom súťažiť cez LAN.

#### **Príchod internetu a online multiplayeru (90. roky)**

S rozvojom internetu sa multiplayerové hry presunuli z LAN na online priestor:

* **Quake (1996)**: Tvorcovia hry Quake, id Software, integrovali podporu pre hranie cez internet. Táto FPS hra sa stala základom pre moderný eSports a bola prvou hrou, ktorá zaviedla koncept online zápasov medzi hráčmi na celom svete.
* **Ultima Online (1997)**: Táto hra od spoločnosti Origin Systems bola jednou z prvých úspešných MMORPG, ktorá umožnila tisícom hráčov spolupracovať a súťažiť v spoločnom virtuálnom svete.
* **StarCraft (1998)**: Táto real-time strategická hra od Blizzard Entertainment priniesla kompetitívny online multiplayer, ktorý bol populárny najmä v Južnej Kórei a položil základy pre moderný eSports.

#### **Moderné multiplayerové hry (2000 – súčasnosť)**

Od 2000-tych rokov sa multiplayerové hry stali základom herného priemyslu:

* **World of Warcraft (2004):** MMORPG od Blizzard Entertainment priniesla masové online hranie miliónom hráčov.
* **Call of Duty a Battlefield:** Tieto FPS série definovali multiplayerové zážitky s pokročilými online funkciami, ako sú tímy, ranky a globálne súťaže.
* **Fortnite a Battle Royale:** Fortnite od Epic Games (2017) a ďalšie battle royale hry (napr. PUBG) zaviedli nový žáner, ktorý spája sociálnu interakciu, súťaženie a obrovské počty hráčov na jednom serveri.
* **Among Us (2020):** Táto hra ukázala, že jednoduché koncepty môžu spájať milióny hráčov v zábavnom a sociálnom zážitku.

#### **Efekt na herný priemysel**

Multiplayerové hry mali obrovský dopad na herný priemysel:

* **Ekonomický rast:** Multiplayerové hry generujú miliardy dolárov prostredníctvom predaja hier, mikrotransakcií a predplatného.
* **Komunity:** Multiplayerové hry vytvorili globálne komunity hráčov, ktoré súperia, spolupracujú a socializujú sa.
* **Vznik eSports:** Multiplayerové hry ako League of Legends, Dota 2 a CS:GO položili základy profesionálneho hrania, ktoré sa stalo celosvetovým fenoménom.

Záverom, história multiplayerových hier ukazuje, ako sa technológie a kreativita spojili, aby vytvorili nové formy interakcie. Multiplayerové hry nie sú len o zábave, ale aj o budovaní komunít a spoločných zážitkov.

### Úvod

Počas štúdia na odbore PDT som sa naučil pracovať a orientovať v rôznych herných enginoch ako sú Unity a Unreal Engine.

Cieľom práce bolo je naučiť sa o softvérovej architektúre v počítačových hrách, ktorú uplatním na projekte, a úspešne posielať informácie medzi dvoma počítačmi v reálnom čase.

### 

### Server

Server je softvér alebo počítač, ktorý v umožňuje komunikáciu medzi počítačmi. Komunikácia funguje pomocou paketov. Paket (ang. packet - balíček) je ucelený blok dát, ktorý sa v sieti s programovaním paketov prenáša a spracováva ako celok.

### Dedicated server

Server bude slúžiť ako sprostredkovateľ medzi hráčmi. Hráč vytvorí spojenie iba so serverom, posielať a prijímať dáta môže iba od serveru. Ak bude hráč chcieť vykonať akciu ako je pohyb alebo výstrel bude musieť vytvoriť paketu, ktorú pošle serveru. Server sa následne spracuje paketu a rozhodne či môže byť akcia vykonaná a ak áno tak informuje všetkých pripojených hráčov o akcií, ktorú hráč vykonal. Softvér pri tom nebude vykreslovať žiadnu grafiku. Server bude čisto príkazový riadok do ktorého sa budú vypisovať upozornenia o informácie o dianí v simulácií, čo bude použité na debugovanie.

### Peer to peer

Oficiálne sa pod peer to peer rozumie to, že počítače sú prepojené medzi sebou bez centralizovanej jednotky a informácie si posielajú medzi sebou. V kontexte multiplayerových hier sa pod peer to peer väčšinou rozumie skôr, že server je hráč, ktorý funguje zároveň ako server. Má všetky vlastnosti dedikovaného serveru akorát môže spracovávať vlastné vstupy a posielať ich ostatným hráčom. Namiesto konzolového riadku, hra vykresluje normálnu grafiku.

### Paket systém

Pomocou knižnici System.Net.Sockets vieme posielať dáta medzi počítačmi. Server ale potrebuje rozoznať a interpretovať tieto dáta a správne ich spracovať. Musíme vytvoriť univerzálny formát paketu, podľa ktorého sa môže server riadiť.

**Formát paketu:**

1. Id - reprezentovaný bude bytom (číslom od 0 do 255), toto číslo server použije na vyhľadávanie v zozname návodov na interpretáciu obsahu
2. Dĺžka- reprezentovaný bude číslom, ktorý určuje akú dĺžku majú všetky informácie v pakete dokopy v bytoch
3. Obsah- obsah paketu bude vpísaný nakoniec

Server vie teraz rozoznať kde začína a končí paket, nevie ale čo má s obsahom paketu urobiť. Serveru môžeme urobiť zoznam návodov, v ktorom bude vyhľadávať pomocou Id paketu. Ak Id v zozname nenájde tak paket ignoruje. Týmto spôsobom môžeme urobiť aj vytváranie paketu, kde server dá informácie návodu a ten ho spracuje do zoznamu bytov, ktorý vie server poslať v pakete hráčom.

**Formát návodu:**

1. Id - reprezentovaný bude bytom (číslom od 0 do 255)
2. Write - funkcia, ktorá konvertuje informácie do poslateľnej formy
3. Handle - funkcia, ktorá interpretuje obsah pakety

### Zoznam paketov

**Zoznam paketov, ktoré môže poslať server**

1. Privítací paket - server pošle tento paket klientovi keď sa pripojí na server. Slúži na potvrdenie pripojenia od klienta. Paket obsahuje Id klienta
2. Spawnovací paket - server pošle tento paket každému pripojenému klientovi. Po príjmutí si klient, spawne tank pripojeného hráča. Tento paket obsahuje Id klienta, ktorý sa pripojil, jeho používateľské meno a pozíciu, na ktorej sa má spawnut tank hráča.
3. Zmena pozície hráča - server pošle tento paket každému pripojenému klientovi, podľa inputu od klienta. Paket obsahuje Id klienta, ktorý sa pohol, jeho aktualizovanú pozíciu vo svete a jeho aktualizovanú orientáciu.
4. Streľba - server pošle tento paket, ak sa klient pokusí vystreliť v hre. Server predtým vypočíta či streľba niekoho trafila a ak áno pošle následne paket o smrti daného hráča
5. smrť hráča - server pošle tento paket každému pripojenému klientovi, že daný hráč prehral dané kolo. Daný hráč zmizne z herného pola a hrajú zvyšný hráči.
6. Odpojenie - Ak server detekuje, že jeden z klientov stratil pripojenie na server. Pošle tento paket ostatným klientom aby mohli vymazať inštanciu tanku odpojeného hráča.

**Zoznam paketov, ktoré môže poslať klient**

1. Prijímanie privítacieho paketu - klient pošle tento paket ak úspešne príjme privítací paket od servera. Tento paket obsahuje používateľské meno klienta.
2. Vstup klienta - tento paket sa pošle keď hráč stlačí, klávesy zodpovedajúcemu ovládania tanku. Paket obsahuje Id klienta a zabalený byte. Do bytu sa uloží každý vstup od hráča, ktorý je stlačený počas posielania tohto paketu. Bit sa zmení na jeden ak hráč vykonáva túto akciu a nula ak nie. Na prvom mieste

### Zabezpečenie paketov

Hráčom sa nedá veriť. Podvádzanie (ang. cheating - podvádzanie, hacking - hackovanie) je keď hráč porušuje pravidlá alebo štandardy hry. Hráčovi nič nebráni aby poslal paket s upravenými informáciami pomocou externého programu. Tento problém má v súčasnosti množstvo najznámejších multiplayerových hier.

### Server Autoritatívny model

Jedno riešenie na podvádzanie je limitovanie typov paketu, ktoré môže hráč poslať. Každý paket je označený číslom od 0 do 255 (bytom), podľa ktorého sa server rozhodne ako informácie v pakete spracovať. Ak server prečíta paket s číslom, ktoré server nevie rozpoznať tak sa paket ignoruje.

Problém je ak klient pošle paket s upravenými informáciami, ktoré bude vedieť prečítať. Ak máme paket, ktorý ma klient poslať ak sa posunie s postavou hráča, ktorý obsahuje pozíciu hráčovej postavy v priestore hry, môže ho klient upraviť a posunúť svoju postavu na ľubovoľnú pozíciu v hre. Server by paket ale prečítal normálne čo by klientovi umožnilo podvádzať a obchádzať obmedzenia v pohybovaní postáv.

Riešením je zmeniť dáta, ktoré má paket obsahovať. Namiesto pozície hráča v hernom priestore, bude paket obsahovať smer, v ktorom sa chce hráč hýbať. V tomto prípade bude môcť klient upraviť len smer, v ktorom sa chce hýbať a server sa postará o zmenu pozície, ktoré pošle všetkým klientom. Server ma vďaka tomu pod kontrolu pozíciu hráča a zabraňuje podvádzaniu klientovi aspoň čo sa týka pohybu. Tento model sa volá Server-Authoritative, lebo server ma finálne slovo a správa sa ako rozhodca v priebehu hry: Každú akciu, ktorú chce klient spraviť, musí server potvrdiť predtým ako sa pošle ostatným hráčom.

### 

### Predikcia klientov

Používaním servero-autoritatívneho modelu sa nám naskytne nový problém. Každý paket má oneskorenie medzi tým ako ho klient pošle a keď ho server príjme. Toto je problém internetu a okrem zníženia geografickej vzdialenosti medzi serverom a klientmi sa oneskorenie nedá vyriešiť. Problém nastáva ak bude musieť server potvrdiť každý paket od klientov. Náš paket, ktorý obsahuje smer, v ktorom sa chce hráč posunúť, bude mať oneskorenie medzi tým ako hráč stlačí tlačidlo na pohyb vpred a keď prijme paket od serveru, ktorý potvrdzuje pohyb vpred. Toto oneskorenie si hráč ľahko všimne, najmä ak sa nachádza ďaleko od servera.

Tento problém vyriešíme tým, že klient bude simulovať svoj pohyb, zároveň ako posiela pakety pohybu na server. Server potom simuluje pohyb a pošle pakety o zmene pozície daného hráča. Keď hráč, ktorý sa pohol príjme paket od servera o svojom pohybe, porovná aktualizovanú pozíciu so svojou aktuálnou. Ak je vzdialenosť medzi lokálnou a serverovou pozíciou vyšší ako danný limit. Hráčovu pozíciu zmeníme na tú definovanú serverom. Klient týmto predvída aký bude výsledok svojho pohybu. Hráč má teraz hladký zážitok a pri tom má server stále konečné rozhodnutie o pozícii hráča.

**Interpolácia pohybu**

Základné klient prijíma pakety 30 krát za sekundu. Tento, na ktorom klient prijíma pakety sa volá Tickrate. Ak by boli pozície hráčov vykreslované v tomto intervale, tak by pohyb vyzeral sekavo a zážitok hráča by sa zhoršil. Stratené pakety by mohli tiež predstavovať problém. Hráčova postava by potom skákala kade tade po obrazovke. Trik je v tom aby sme vykreslovanú pozíciu hráča priebežne interpolovali medzi poslednými dvoma pozíciami poslanými serverom. Interpolácia znamená, v numerickej matematike, hladanie približnej hodnoty funkcie v intervale, ak je hodnota známa na inych bodoch daného intervalu. Hľadáme teda pozíciu medzi dvoma bodmi. Prvý bod je začiatok a je to predchádzajúci paket s pozíciou hráča, druhý bod je koniec a je to momentálny paket s pozíciou hráča. Vykreslovanú pozíciu hráča meníme medzi danými bodmi podľa ubehnutého času. Ak paket príde cca. každých 33,3 stotiny sekundy, tak budeme vidieť hráčovu pozíciu vždy 33,3 stotiny sekundy pozadu.